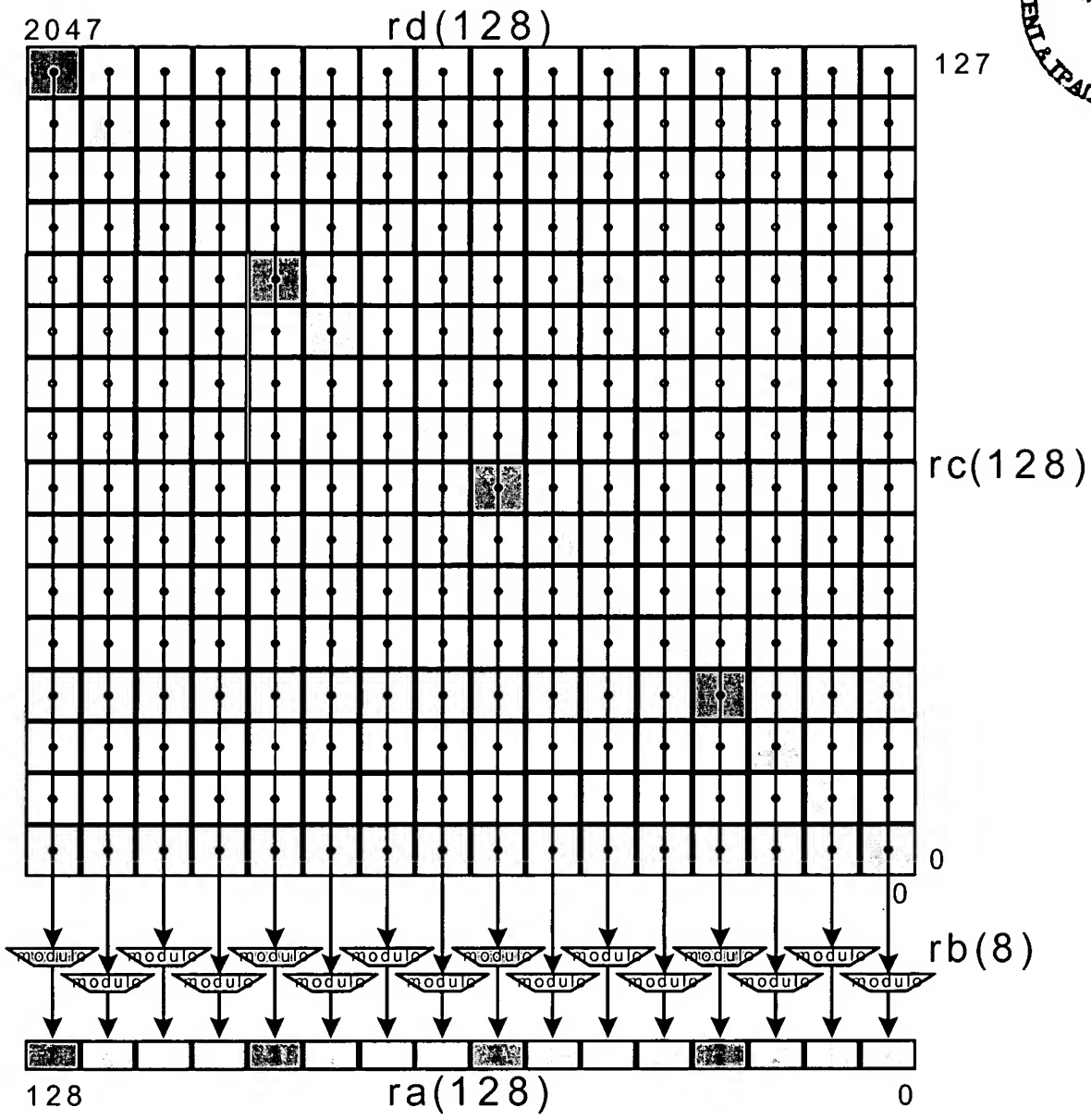
## Definition

```
def GroupTernary(op,size,rd,rc,rb,ra) as
      d ← RegRead(rd, 128)
      c ← RegRead(rc, 128)
      b ← RegRead(rb, 128)
      case op of
            G.MUX:
                  a ← (c and d) or (b and not d)
      endcase
      RegWrite(ra, 128, a)
enddef
```

## Exceptions

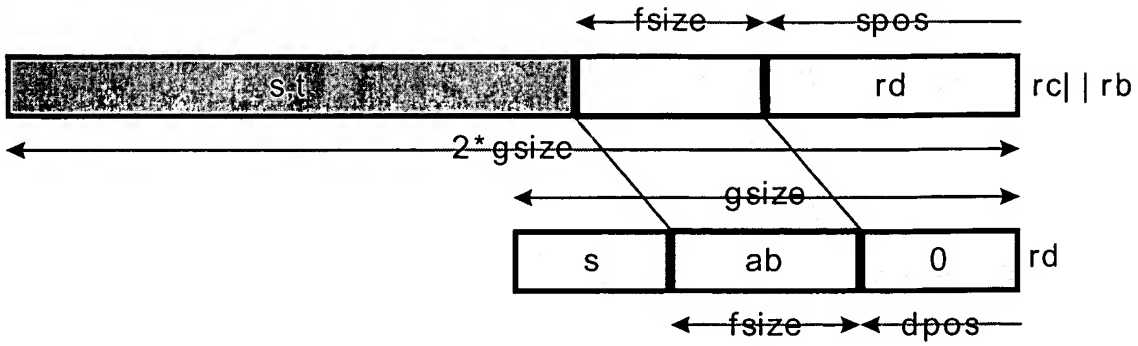
none

**Fig. 31E**



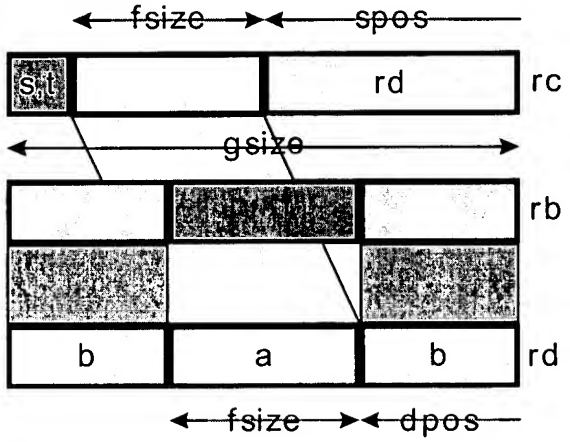

**Ensemble multiply Galois field bytes**

**Fig. 42D**



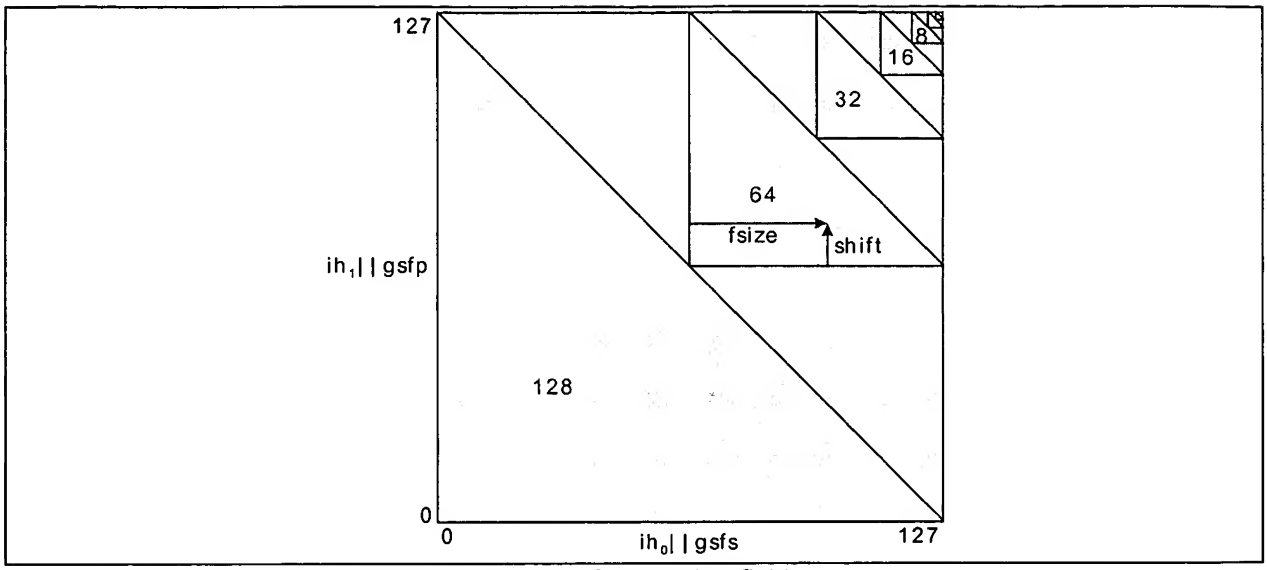

**Crossbar extract**

**Fig. 44C**



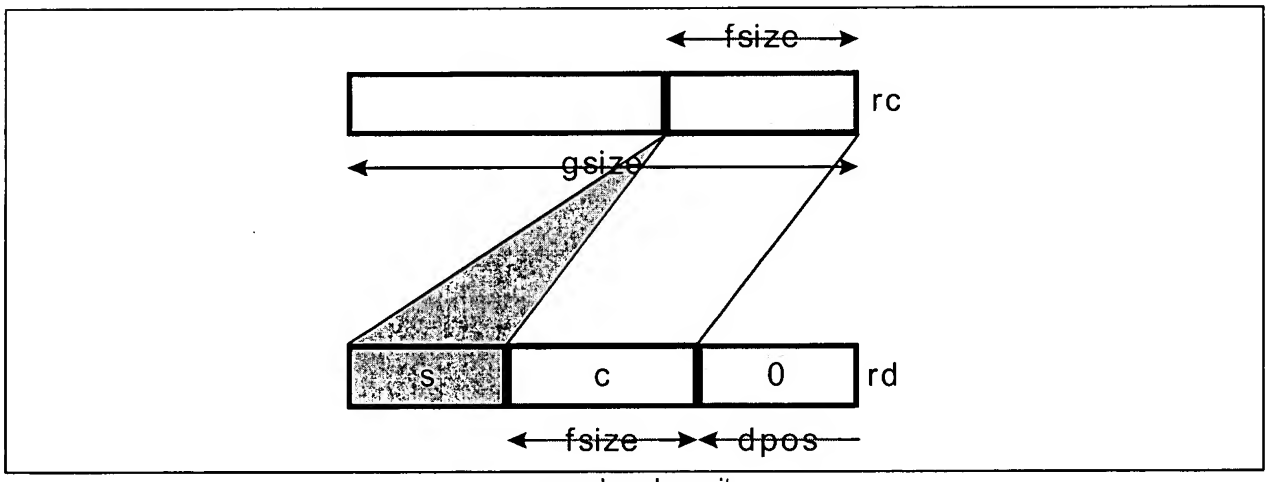

**Crossbar merge extract**

**Fig. 44D**




encoding for crossbar field

**Fig. 45D**




crossbar deposit

**Fig. 45E**

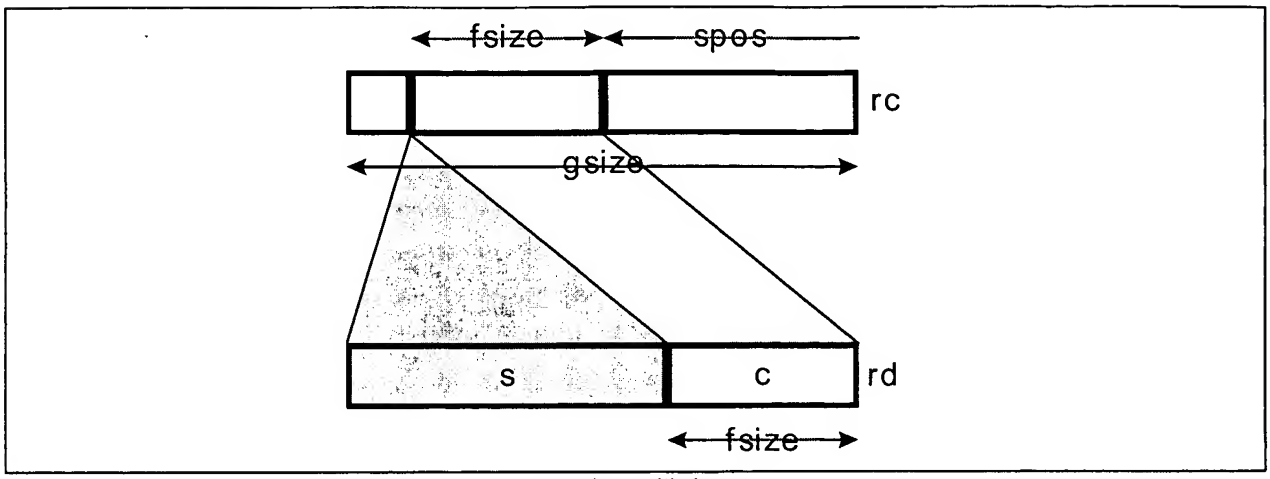



**Fig. 45F**

**Operation codes**

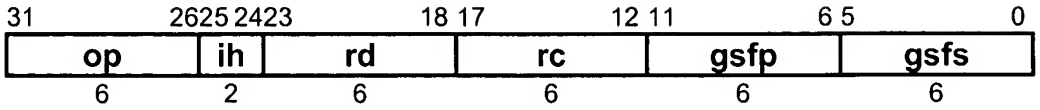
| | |
|---|---|
| X.DEPOSIT.M.2 | Crossbar deposit merge pecks |
| X.DEPOSIT.M.4 | Crossbar deposit merge nibbles |
| X.DEPOSIT.M.8 | Crossbar deposit merge bytes |
| X.DEPOSIT.M.16 | Crossbar deposit merge doublets |
| X.DEPOSIT.M.32 | Crossbar deposit merge quadlets |
| X.DEPOSIT.M.64 | Crossbar deposit merge octlets |
| X.DEPOSIT.M.128 | Crossbar deposit merge hexlet |

**Fig 45G**

**Format**

X.op.gsize rd@rc,isize,ishift

rd=xopgsize(rd,rc,isize,ishift)




assert isize+ishift ≤ gsize
assert isize≥1
$ih_0$ || gsfs ← 128-gsize+isize-1
$ih_1$ || gsfp ← 128-gsize+ishift

**Fig 45H**

## Definition

```
def CrossbarFieldInplace(op,rd,rc,gsfp,gsfs) as
    c ← RegRead(rc, 128)
    d ← RegRead(rd, 128)
    case ((op1 || gsfp) and (op0 || gsfs)) of
        0..63:
            gsize ← 128
        64..95:
            gsize ← 64
        96..111:
            gsize ← 32
        112..119:
            gsize ← 16
        120..123:
            gsize ← 8
        124..125:
            gsize ← 4
        126:
            gsize ← 2
        127:
            raise ReservedInstruction
    endcase
    ishift ← (op1 || gsfp) and (gsize-1)
    isize ← ((op0 || gsfs) and (gsize-1))+1
    if (ishift+isize>gsize)
        raise ReservedInstruction
    endif
    for i ← 0 to 128-gsize by gsize
        a[i+gsize-1..i] ← d[i+gsize-1..i+isize+ishift] || c[i+isize-1..i] || d[i+ishift-1..i]
    endfor
    RegWrite(rd, 128, a)
enddef
```

## Exceptions

Reserved instruction

**Fig 45l**

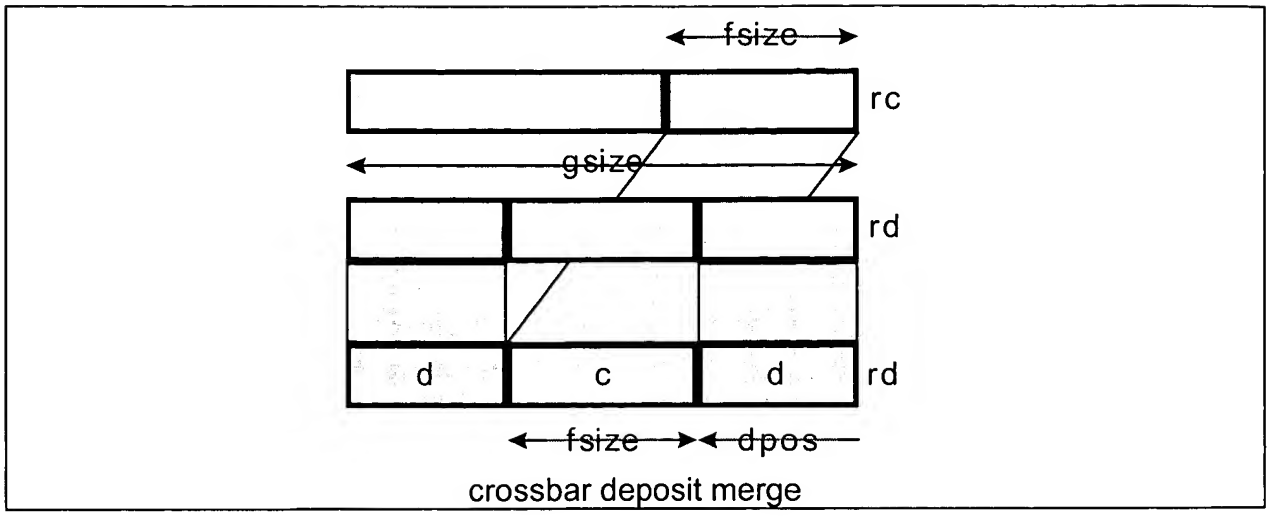



**Fig 45J**

**Operation codes**

| | |
|---|---|
| G.MUX | Group multiplex |

**Redundancies**

| | | |
|---|---|---|
| G.MUX ra=rd,rc,rc | ⇔ | *G.COPY ra=rc* |
| G.MUX ra=ra,rc,rb | ⇔ | G.BOOLEAN ra@rc,rb,0x11001010 |
| G.MUX ra=rd,ra,rb | ⇔ | G.BOOLEAN ra@rd,rb,0x11100010 |
| G.MUX ra=rd,rc,ra | ⇔ | G.BOOLEAN ra@rd,rc,0x11011000 |
| G.MUX ra=rd,rd,rb | ⇔ | G.OR ra=rd,rb |
| G.MUX ra=rd,rc,rd | ⇔ | G.AND ra=rd,rc |

**Format**

G.MUX ra=rd,rc,rb

ra=gmux(rd,rc,rb)

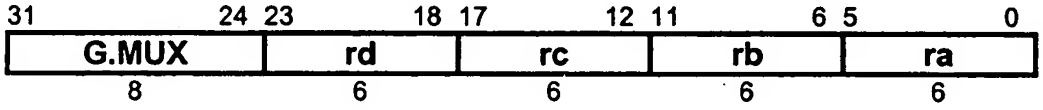



**Fig. 31D**